***Listing of the Claims:***

1. (Currently Amended) A system for decoupling commercial-off-the-shelf software applications from data stores, the system comprising:

a plurality of commercial-off-the-shelf software applications each ~~operable~~ compatible with one of a plurality of first data stores, each of the plurality of commercial-off-the-shelf software applications ~~providing output~~ submits a data request compatible with one of the plurality of first data stores;

a plurality of second data stores;

a plurality of drivers, wherein each of the plurality of first data stores and the plurality of second data stores has a corresponding one of the plurality of drivers configured to receive the data request and pass the data request to the corresponding data store;

at least one processor;

a listener, recorded on a computer readable medium, when executed by [[the]] at least one processor, ~~to receive~~ simulates one of the plurality of drivers corresponding with one of the plurality of first data stores and receives the ~~output~~ data request from one of the plurality of commercial-off-the-shelf software applications that is compatible with the one of the plurality of first data stores simulated by the listener;

a translator, recorded on a computer readable medium, in communication with the listener and the plurality of second data stores, the translator, when executed by [[the]] at least one processor, [[to]] receives the ~~output~~ data request from the listener, ~~and configured to~~ translates the ~~output~~ data

request, and submits the translated data request for one of the plurality of drivers corresponding with one of the plurality of second data stores for storage by the one of the plurality of second data stores.

2. (Currently Amended) The system of Claim 1, wherein the translator translates the data request into a generic format, and further comprising a data access layer, recorded on a computer readable medium, in communication with the translator and, when executed by [[the]] at least one processor, [[to]] determines [[where]] to direct the ~~output~~ data request from one of the commercial-off-the-shelf software applications to the one of the plurality of second data stores, and translates the translated data request from the generic format into a storage format of the one of the plurality of second data stores.

3. (Currently Amended) The system of Claim 2, wherein the data access layer maintains an enterprise data model including a data map of where to direct the ~~output~~ data request of each of the commercial-off-the-shelf software applications.

4. (Currently Amended) The system of Claim 3, wherein the data access layer receives the translated ~~output~~ data request from the translator and directs the translated ~~output~~ data request to one of the plurality of second data stores.

5. (Currently Amended) The system of Claim 2 [[1]], wherein a first commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications ~~provides~~ submits a first ~~output~~ data request in a first relational database format and wherein the ~~translator translates~~ data access layer translates the first ~~output~~ data request to a second relational database format.

6. (Currently Amended) The system of Claim 5, wherein a second commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications ~~provides~~ submits a second ~~output~~ data request in an older version of the first relational database format and wherein the ~~translator translates~~ data access layer translates the second ~~output~~ data request to a newer version of the first relational database format.

7. (Currently Amended) The system of Claim 2 [[1]], wherein a first commercial-off-the-shelf software application of the plurality of commercial-off-the-shelf software applications ~~provides~~ submits a first ~~output~~ data request in an older version of a first relational database format and wherein the ~~translator translates~~ data access layer translates the first ~~output~~ data request to a newer version of the first relational database format.

8. (Currently Amended) The system of Claim 1, wherein at least one of the second data stores ~~is associated~~ corresponds with one of the plurality of first data stores.

9. (Original) The system of Claim 8, wherein the at least one of the second data stores is further defined as a newer version data store of one of the plurality of first data stores.

10. (Previously Presented) The system of Claim 9, wherein at least one of the second data stores is further defined as a newer version of a relational database of a first vendor and wherein one of the plurality of first data stores is further defined as an older version of the relational database of the first vendor.

11. (Previously Presented) The system of Claim 9, wherein at least one of the second data stores is further defined as a newer version of a relational database of a second vendor and wherein one of the plurality of first data stores is further defined as an older version of the relational database of the second vendor.

12. (Currently Amended) The system of Claim 1, wherein the plurality of commercial-off-the-shelf software applications are each operable with only one of a plurality of data stores, each of the plurality of commercial-off-the-shelf software applications ~~providing output~~ submitting data requests compatible with only one of the plurality of data stores.

13. (Currently Amended) A system for maintaining compatibility of commercial-off-the-shelf software applications with data stores, the system comprising:

a commercial-off-the-shelf software application operable with only a first data store, the commercial-off-the-shelf software application ~~providing an output~~ submits a data request compatible with only the first data store;

a first driver configured to receive the data request and pass the data request to the first data store;

at least one processor;

a listener, recorded on a computer readable medium, when executed by [[the]] at least one processor, ~~to receive~~ simulates the first driver and receives the ~~output~~ data request from the commercial-off-the-shelf software application submitted for the first driver;

a translator, recorded on the computer readable medium, in communication with the listener, when executed by [[the]] at least one processor, [[to]] receives the ~~output~~ data request from the listener and ~~configured to~~ translates the ~~output~~ data request into a generic format to produce a first translated data request;

a data access layer, recorded on the computer readable medium, in communication with the translator and, when executed by [[the]] at least one processor, [[to]] determines, based on an enterprise data model, ~~where~~ to direct the ~~output~~ data request of the commercial-off-the-shelf software applications to a second data store and translates the first

translated data request from the generic format into a storage format of the second data store to produce a second translated data request;

a wrapper, recorded on the computer readable medium, when executed by [[the]] at least one processor, [[to]] receives the second translated ~~output~~ data request from the data access layer and [[to]] wraps the second translated ~~output~~ data request based on [[a]] the storage format of the~~; and a~~, second data store ~~based on the storage format and configured to receive and store the wrapped and translated output.~~

a second driver configured to receive the wrapped second translated data request and pass the wrapped second translated data request to the second data store; and

the second data store receives the wrapped second translated data request from the second driver and performs an action specified in the data request.

14. (Previously Presented) The system of Claim 13, wherein the second data store is one of a newer version data store of the first data store and a different vendor database than the first data store.

15. (Cancelled)

16. (Currently Amended) A system for integration of commercial-off-the-shelf software applications and databases, the system comprising:

a commercial-off-the-shelf software application operable with a first data store, the commercial-off-the-shelf software application ~~providing an output~~ submits a data request compatible with the first data store;

a first driver configured to receive the data request and pass the data request to the first data store;

at least one processor;

a listener, recorded on a computer readable medium, when executed by [[the]] at least one processor, ~~to receive~~ simulates the first driver and receives the ~~output~~ data request from the commercial-off-the-shelf software application submitted for the first driver;

a translator, recorded on a computer readable medium, in communication with the listener, when executed by [[the]] at least one processor, [[to]] receives the ~~output~~ data request from the listener and ~~configured to~~ translates the ~~output~~ data request;

a second driver configured to receive the translated data request and pass the translated data request to a second data store; [[a]] wherein the second data store ~~operable to~~ receives ~~and store~~ the translated ~~output~~ data request from the second driver and performs an action specified in the data request; and

a service broker, recorded on the computer readable medium, when executed by [[the]] at least one processor, [[to]] maintains a record

of ~~transaction output~~ data requests from the commercial-off-the-shelf software application and stored in the second data store, the service broker further configured to roll-back failed ~~transactions~~ data requests.

17. (Currently Amended) The system of Claim 16, wherein the translator translates the data request into a generic format, and further comprising a data access layer, recorded on a computer readable medium, in communication with the translator and, when executed by [[the]] at least one processor, [[to]] determines, based on an enterprise data model, [[where]] to direct the ~~output~~ data request from one of the commercial-off-the-shelf software applications to the second data store, and translates the translated data request from the generic format into a storage format of the second data store.

18.( Currently Amended) The system of Claim 16, wherein the commercial-off-the-shelf software application is operable with only the first data store, and wherein the commercial-off-the-shelf software application ~~provides~~ submits the ~~output~~ data request compatible with only the first data store.

19. (Currently Amended) The system of Claim 16, wherein the service broker further comprises:

a transaction data store configured [[to]] that maintains a record of the ~~output~~ data request by the commercial-off-the-shelf software application;

an exception handler ~~configured to~~ that identif[[y]]ies a failed transaction and communicates with the transaction data store to restore the second data store to a state prior to the failed transaction.

20. (Currently Amended) The system of Claim 19, further comprising a data warehouse, recorded on the computer readable medium, and wherein the data warehouse, when executed by [[the]] at least one processor, is asynchronously updated with the ~~output~~ data request from the commercial-off-the-shelf software application.

21. (Original) The system of Claim 19, wherein a compensating transaction is used to restore the failed transaction.

22. (Original) The system of Claim 21, wherein an XA transaction is used in combination with the compensating transaction to restore the failed transaction.

23. (Currently Amended) The system of Claim 19, further comprising:

a data warehouse, recorded on the computer readable medium, when executed by [[the]] at least one processor, [[to]] maintains data;

a query processor, recorded on the computer readable medium, when executed by [[the]] at least one processor, [[to]] manages transaction processing of data requests from the commercial-off-the-shelf software application; and

a metadata repository, recorded on the computer readable medium, when executed by [[the]] at least one processor, [[to]] maintains a logical data model related to the data, wherein the metadata repository instructs the query processor regarding handling of the data requests from the commercial-off-the-shelf software application and between the second data store and the data warehouse.